IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appl. No.:	10/037,683	§	Confirmation No.:	8294
Applicant:	Robert S. Brayton	§		
Filed:	01/04/2002	§		
TC/A.U.:	2178	§		
Examiner:	Adam L. Basehoar	§		
Title:	METHOD TO SERVE REAL-TIME DATA IN EMBEDDED WEB SERVER	§		
Docket No.:	200302369-1 (HPC.0864US)	§		

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

The final rejection of claims 29-47 is hereby appealed.

I. REAL PARTY IN INTEREST

The real party in interest is the Hewlett-Packard Development Company, LP. The Hewlett-Packard Development Company, LP, a limited partnership established under the laws of the State of Texas and having a principal place of business at 11445 Compaq Center Drive West, Houston, TX 77070, U.S.A. (hereinafter "HPDC"). HPDC is a Texas limited partnership and is a wholly-owned affiliate of Hewlett-Packard Company, a Delaware Corporation, headquartered in Palo Alto, CA. The general or managing partner of HPDC is HPQ Holdings, LLC.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF THE CLAIMS

Claims 29-47 have been finally rejected and are the subject of this appeal. Claims 1-28 and 48 have been cancelled.

IV. STATUS OF AMENDMENTS

No amendment after the final rejection of December 2, 2009 has been submitted. Therefore, all amendments have been entered.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The following provides a concise explanation of the subject matter defined in each of the independent claims involved in the appeal, referring to the specification by page and line number and to the drawings by reference characters, as required by 37 C.F.R. § 41.37(c)(1)(v). Each element of the claims is identified by a corresponding reference to the specification and drawings where applicable. Note that the citation to passages in the specification and drawings for each claim element does not imply that the limitations from the specification and drawings should be read into the corresponding claim element. Note also that the cited passages are provided as examples, as other passages in the specification or drawings not cited may also be relevant to the corresponding claim elements.

Independent claim 29 recites a method of serving data from a management module (Fig. 2:106; Spec., p. 7, ln. 17 – p. 8, ln. 3) of a managed server, comprising:

serving a web page (Fig. 2:114) to a requesting computer (Fig. 2:102) from the managed server (Fig. 2:104), the web page comprising a source call to an object file and code including scripting functions defined by the object file, wherein the requesting computer is remote from the managed server, and wherein at least one of the scripting functions is for merging data associated with the object file with the web page (Spec., p. 8, ln. 7 – p. 9, ln. 24; p. 9, ln. 29 – p. 10, ln. 6);

receiving (Fig. 3:302) a request from the requesting computer at the managed server for the object file, wherein the request is received after the web page has been served to the requesting computer and after the requesting computer has evaluated the at least one scripting function (Spec., p. 10, ln. 8-10; p. 13, ln. 13-15);

populating (Fig. 4:308) the object file in real-time with data from the management module of the managed server after both serving the web page and receiving the request for the object file (Spec., p. 5, ln. 18-19; p. 10, ln. 10-11; p. 13, ln. 20 - p. 14, ln. 7); and

serving (Fig. 4:312) the object file to the requesting computer after populating the object file (Spec., p. 5, ln. 18-19; p. 10, ln. 13-16; p. 14, ln. 1-2).

Independent claim 37 recites a method of displaying a web page, comprising:

requesting (Fig. 3:202) at least a frame of a web page (Fig. 2:114) from a managed server (Fig. 2:104), wherein the frame comprises a first embedded object and a call to a scripting language function defined by the first embedded object, wherein the scripting language function is for merging data corresponding to the first embedded object with the web page (Spec., p. 8, ln. 7 – p. 9, ln. 24; p. 9, ln. 29 – p. 10, ln. 6);

receiving (Fig. 3:204) the frame from the managed server (Spec., p. 11, ln. 9-11);

based on evaluating (Fig. 3:206) the scripting language function, requesting, by a requesting (Fig. 3:208) computer (Fig. 2:102), the data corresponding to the first embedded object from the managed server after receiving the frame from the managed server (Spec., p. 11, ln. 13-20);

receiving (Fig. 3:212), by the requesting computer, the data corresponding to the first embedded object (Spec., p. 12, ln. 5-6);

calling, by the requesting computer, the scripting language function defined by the first embedded object (Spec., p. 5, ln. 20-21; p. 8, ln. 24-25; p. 9, ln. 16-24); and

merging (Fig. 3:214), by the requesting computer, the data corresponding to the first embedded object into the frame (Spec., p. 12, ln. 6-13).

Independent claim 46 recites a server, comprising:

a management module (Fig. 2:106) configured to generate dynamic data (Spec., p. 7, ln. 19 – p. 8, ln. 3); and

a file system configured to store a web page (Fig. 2:114) that has both a first embedded object configured to access the dynamic data and a second embedded object configured to merge the dynamic data with the web page, wherein the first embedded object is executable on a client remote from the server to request the dynamic data from the server, and wherein the web page includes a scripting language function defined by the second embedded object, the scripting language function for merging the dynamic data with the web page (Spec., p. 8, ln. 7 – p. 9, ln. 24; p. 9, ln. 29 – p. 10, ln. 6);

wherein the server (Fig. 2:104) is configured to further:

send, to the client, the web page that has the first embedded object, the second embedded object, and the scripting language function defined by the second embedded object (Spec., p. 10, ln. 8-10; p. 13, ln. 13-15);

after sending the web page, receive a request from the client that is based on evaluating the scripting language function of the web page at the client (Spec., p. 5, ln. 18-19; p. 10, ln. 13-16; p. 14, ln. 1-2);

in response to the request, retrieve the dynamic data and send the retrieved dynamic data to the client for merging with the web page (Spec., p. 5, ln. 18-19; p. 10, ln. 10-11; p. 13, ln. 20 - p. 14, ln. 7).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL[1]

A. Claims 29-43, 46, and 47 were rejected under 35 U.S.C. § 103(a) as unpatentable over Mateos (U.S. Patent Publication No. 2003/0050995) in view of Coates (U.S. Patent No. 6,952,737).

B. Claim 44 was rejected under 35 U.S.C. § 103(a) as unpatentable over Mateos in view of Coates, and further in view of Chen (U.S. Patent No. 6,021,437).

C. Claim 45 was rejected under 35 U.S.C. § 103(a) as unpatentable over Mateos in view of Coates, and further in view of Lynch (U.S. Patent No. 6,823,319).

VII. ARGUMENT

The claims do not stand or fall together. Instead, Appellant presents separate arguments for various independent and dependent claims. Each of these arguments is separately argued below and presented with separate headings and sub-headings as required by 37 C.F.R. § 41.37(c)(1)(vii).

A. Claims 29-43, 46, and 47 were rejected under 35 U.S.C. § 103(a) as unpatentable over Mateos (U.S. Patent Publication No. 2003/0050995) in view of Coates (U.S. Patent No. 6,952,737).

1. Claims 37-40, 42, 43.

It is respectfully submitted that independent claim 37 is non-obvious over Mateos in view of Coates.

To make a determination under 35 U.S.C. § 103, several basic factual inquiries must be performed, including determining the scope and content of the prior art, and ascertaining the differences between the prior art and the claims at issue. *Graham v. John Deere Co.*, 383 U.S. 1, 17, 148 U.S.P.Q. 459 (1965). Moreover, as held by the U.S. Supreme Court, it is important to identify a reason that would have prompted a person of ordinary skill in the art to combine

[1] Page 13 of the Final Office Action rejected claim 48 over Mateos, Coates, and Compaq. However, claim 48 has been cancelled without prejudice and therefore the rejection of claim 48 is not addressed in this Appeal Brief.

reference teachings in the manner that the claimed invention does. *KSR International Co. v. Teleflex, Inc.*, 127 S. Ct. 1727, 1741, 82 U.S.P.Q.2d 1385 (2007).

Here, claim 37 recites a method of displaying a web page comprising:

- requesting at least a frame of a web page from a managed server, wherein the frame comprises a first embedded object and a call to a scripting language function defined by the first embedded object, wherein the scripting language function is for merging data corresponding to the first embedded object with the web page;
- receiving the frame from the managed server;
- based on evaluating the scripting language function, requesting, by a requesting computer, the data corresponding to the first embedded object from the managed server after receiving the frame from the managed server;
- receiving, by the requesting computer, the data corresponding to the first embedded object;
- calling, by the requesting computer, the scripting language function defined by the first embedded object; and
- merging, by the requesting computer, the data corresponding to the first embedded object into the frame.

Note that requesting the data corresponding to the first embedded object from the managed server is performed **after receiving the frame of a web page from the managed server, where the frame has a first embedded object and a call to a scripting language function defined by the first embedded object**. The Examiner conceded that Mateos fails to disclose the foregoing claimed subject matter. 12/12/2009 Office Action at 5. Instead, the Examiner cited Coates as purportedly disclosing the claimed feature. *Id.* at 5-6.

Specifically, the Examiner cited the following passage of Coates as purportedly disclosing the claimed subject matter: column 26, line 52 – column 27, line 65; Figs. 25-26. *Id.* at 6.

As shown in Figs. 25 and 26, and described by the accompanying text in passages in columns 26 and 27 of Coates cited by the Examiner, an end-user computer sends a request to a client site 2620, and receives, in return, an HTML web page with one or more embedded SRLs

(storage resource locators). Coates, 27:3-6. Using the embedded SRLs, the end-user computer 2610 generates SRL requests directly to the storage center 2650 over a wide area network. *Id.*, 27:6-9. In response, the storage center 2650 serves object files directly to the end-user computer 2610. *Id.*, 27:9-10. The SRLs embedded in the HTML web page are identifiers that allow a user to submit requests directly to the storage center 2650 for object files.

However, although the SRLs allow for the identification of locations of data at the storage center in Coates, the SRLs do not constitute a scripting language function in the web page of claim 37. As specifically recited in claim 37, the scripting language function is for merging data corresponding to the first embedded object with the web page. The SRLs of Coates do not constitute such a scripting language function.

Mateos also clearly fails to disclose or hint at the foregoing subject matter of claim 37. Although Mateos describes use of a CGI program 230 at a web server to build a web page containing data retrieved from a database to a client for display (Mateos, ¶ [0029]), there is absolutely no hint given in Mateos that this web page that is sent to a client for display would include the scripting language function of claim 37, where the scripting language function is for merging data corresponding to the first embedded object with the web page, and **where after receiving a frame that contains a call to the scripting language function, the scripting language function is evaluated, and based on this evaluation, the data corresponding to the first embedded object is requested from the managed server**.

The Response to Arguments section of the Final Office Action further cited ¶¶ [0055]-[0056] of Mateos. 12/02/2010 Office Action at 14. These passages describe the HTML document that represents a web page produced by the CGI program 230 of the server 125s depicted in Fig. 2 of Mateos. Mateos, ¶¶ [0030]-[0054]. As noted above, the web page is

produced by the CGI program 230 at the server and is sent to the client for display. The HTML document that represents such a web page has a data section, which starts with a tag identifying a script. Mateos, ¶ [0055]. The script includes the definition of a table object that consists of a table with two rows. *Id.* A variable for the table object is declared, and dynamic information resulting from a query on the database is assigned to this variable. *Id.*

Paragraph [0056] of Mateos states that the HTML document that represents the web page also includes a view section, which denotes the piece of information to be placed in each cell of a row of a table. *Id.*, ¶ [0056]. The key point of Mateos is that the web page (as represented by the HTML document discussed in ¶¶ [0055]-[0056] of Mateos) is produced by the CGI program 230 at the server, where the CGI program 230 at the server runs queries on a database 250 at the server, as depicted in Fig. 2 of Mateos. The query that is run by the CGI program 230 on the database 250 retrieves dynamic information requested by the user of the client computer. *Id.*, ¶ [0029]. This dynamic information is retrieved into the web page 253 that is represented by the HTML document of ¶¶ [0055]-[0056] of Mateos. Note that the HTML document is then sent to the client computer, where the browser at the client computer interprets the HTML tags discussed in ¶¶ [0055]-[0056] to cause display of a table with the dynamic information retrieved from the database of the server. *Id.*, ¶ [0057]. It is clear that the HTML document is only interpreted or parsed at the client computer, with information for populating the table to be displayed already part of the HTML document. The retrieval of data has already been performed at the server by the CGI program 230 running at the server. There would be absolutely no need whatsoever for the client computer to evaluate the HTML document received from the server, and then based on the evaluation of the HTML document, to further request the data corresponding to an embedded object from the server. Mateos makes it clear that the HTML

document already contains the necessary scripts and data to allow for a table with the dynamic information to be displayed at the client computer upon the client computer receiving the HTML document and parsing the HTML document.

The Response to Arguments section also argued that the Examiner disagrees with Appellant's contention that Coates' SRLs do not constitute a scripting language function in the web page of claim 37, where the scripting language function is for merging data corresponding to the first embedded object with the web page. 12/02/2009 Office Action at 14-15. In response to such argument, the Examiner cited column 27, lines 11-46, of Coates and quoted the following from Coates: "contain content that the client desires to embed in the web page ... downloads the object file to the user." *Id.* at 15. The above quoted language from Coates is taken from two sentences in column 27 of Coates. The first sentence states that the file(s) is (are) file(s) that is (are) requested by the client site, where the file(s) contain(s) content that the client desires to embed in the web page. Coates, 27:12-16. The second sentence is follows: "If the SRL is within the specified time range, then the storage center downloads the object file to the end-user ...." *Id.*, 27:38-40. These two sentences do not support the Examiner's allegation that the SRL constitutes a scripting language function for merging data corresponding to an embedded object with the web page. The SRL of Coates is clearly merely a locator to allow the end-user computer to generate requests directly to the storage center 2650 over a network. *Id.*, 27:6-9. An SRL would be more analogous to a uniform resource locator (URL) that is typically used in the world wide web. Equating the SRL with the scripting language function of claim 37 is erroneous.

The Examiner further referred to another reference, Siow (U.S. Patent No. 6,301,590) as purportedly disclosing Java applets in an HTML page. 12/02/2009 Office Action at 15. This

reference, Siow, is not part of the obviousness rejection of claim 37 based on only Mateos and Coates, and thus citing Siow in the context of the rejection over Mateos and Coates is improper. Moreover, it is clear that Siow does not provide any hint of a web page that is sent to a client for display includes a scripting language function, where the scripting language function is for merging data corresponding to the first embedded object with the web page, and where **after receiving a frame that contains a call to scripting language function, the scripting language function is evaluated, and based on this evaluation, the data corresponding to the first embedded object is requested from the managed server**.

In view of the foregoing, it is clear that even if Mateos and Coates could be hypothetically combined, the hypothetical combination of the references would not have led to the claimed subject matter.

Moreover, in view of the significant differences between the claimed subject matter and the teachings of Mateos and Coates, a person of ordinary skill in the art would clearly not have been prompted to combine the teachings of Mateos and Coates to achieve the claimed subject matter. All that Coates would have hinted to a person of ordinary skill in the art would be that a resource locator, in the form of an SRL, can be provided in a web page that can be used by an end-user computer to request data directly from a storage center. However, a source locator such as the SRL for identifying a location of data has nothing to do with the scripting language function that is for merging data corresponding to the first embedded object with the web page, as recited in claim 37.

Additionally, Mateos relates to a CGI program that is used to construct a web page containing data retrieved from a database, where the web page is sent to the client. However, in Mateos, there is absolutely no hint given whatsoever that the web page that is sent to the user

would contain a scripting language function that would be evaluated for requesting data corresponding to the first embedded object from the managed server, after the web page is received from the managed server.

In fact, Mateos would have led a person of ordinary skill in the art to run the CGI program at the server to build a web page that can be delivered to the client computer for display, without the client computer having to **further request** data based on evaluating the web page. Thus, Mateos would have taught a solution that would have led away from the subject matter of claim 37, which specifically specifies that data corresponding to an embedded object is requested from a managed server after receiving the frame of the web page from the managed server, and based on evaluating the scripting language function contained in such frame of the web page. Therefore, a person of ordinary skill in the art would not have been prompted to combine the teachings of Mateos and Coates to achieve the claimed subject matter.

In view of the foregoing, it is respectfully submitted that the obviousness rejection of claim 37 and its dependent claims over Mateos and Coates is erroneous.

Reversal of the final rejection of the above claims is respectfully requested.

2. Claim 41.

Claim 41 depends from claim 37 and is therefore allowable for at least the same reasons as claim 37. Moreover, claim 41 further recites that the data corresponding to the first embedded object comprises dynamic data from a management module of the managed server. Note that in the context of base claim 37, such dynamic data generated in real-time would be in response to the request made by the requesting computer after receiving the frame of the web page from the managed server and based on evaluating the scripting language function in the frame of the web page from the managed server. Claim 41 further recites that the dynamic data is generated in

real-time in response to the request for the data corresponding to the first embedded object. Again, the request for the data is generated after receiving the frame of the web page from the managed server and based on evaluating the scripting language function contained in the frame of the web page.

With respect to claim 41, the Examiner cited the following passages of Mateos: ¶¶ [0056], [0029], [0057]. 12/02/2009 Office Action at 7. These passages were discussed above in connection with claim 1, and describe how the CGI program 230 running on the server queries the database on the server to retrieve dynamic information requested by the user of the client computer. Mateos, ¶ [0029]. This dynamic information is incorporated into a web page that is then sent to the client computer, where the client computer is able to parse the HTML document representing the web page to display a table containing the dynamic information. *Id.*, ¶ [0057]. The key point of distinction between Mateos and the claimed subject matter is that the web server of Mateos has already populated the web page with the dynamic data that has been retrieved from the database at the server, and this web page is then sent to the client computer for display. Thus, in Mateos, there would be no separate request by the client computer to the managed server for the dynamic data, such that the dynamic data is generated in real-time in response to such request that is made **after** receiving the frame of the web page from the managed server and based on evaluating the scripting language function contained in the frame of the web page.

Claim 41 is therefore further allowable for the foregoing reasons.

Reversal of the final rejection of the above claims is respectfully requested.

3. Claims 29-32, 34-36, 46, 47.

Independent claim 29 is allowable over Mateos and Coates for similar reasons as stated above with respect to claim 37. Specifically, claim 29 recites a method of serving data from a management module of a managed server, comprising:

> serving a web page to a requesting computer from the managed server, the web page comprising a source call to an object file and code including scripting functions defined by the object file, wherein the requesting computer is remote from the managed server, and wherein at least one of the scripting functions is for merging data associated with the object file with the web page;
>
> receiving a request from the requesting computer at the managed server for the object file, wherein the request is received after the web page has been served to the requesting computer and after the requesting computer has evaluated the at least one scripting function;
>
> populating the object file in real-time with data from the management module of the managed server after both serving the web page and receiving the request for the object file; and
>
> serving the object file to the requesting computer after populating the object file.

For reasons similar to those stated above with respect to claim 37, the hypothetical combination of Mateos and Coates would not have led to receiving a request from the requested computer at the managed server for the object file, where the request is received **after the web page has been served to the requesting computer and after the requesting computer has evaluated the at least one scripting function contained in the web page**, where the web page comprises a source call to an object file and code including scripting functions defined by the object file, where at least one of the scripting functions is for merging data associated with the object file with the web page.

Also, for reasons similar to those stated above with respect to claim 37, no reason existed that would have prompted a person of ordinary skill in the art to combine the teachings of Mateos and Coates to achieve the subject matter of claim 29.

Therefore, the obviousness rejection of claim 29 and its dependent claims is clearly erroneous.

Independent claim 46 and its dependent claim 47 is similarly allowable over Mateos and Coates.

Reversal of the final rejection of the above claims is respectfully requested.

4. Claim 33.

Claim 33 depends from claim 29 and is therefore allowable for at least the same reasons as claim 29. Moreover, claim 33 further recites that populating the object file comprises acquiring real-time data indicative of a current status of a server. With respect to claim 33, the Examiner cited the following passages of Mateos: ¶¶ [0003] and [0013]. 12/02/2009 Office Action at 10. These cited passages of Mateos notes that a client computer can interact dynamically with a server computer to download and display information whose content changes each time. However, there is absolutely no hint in either of these passages, or anywhere else in Mateos or in Coates, of populating an object file that comprises acquiring real-time data indicative of a **current status** of a server. The content of information in the server that changes over time is not the same as information that is indicative of a current status of a server, as claimed. Therefore, claim 33 is further allowable over Mateos and Coates for the foregoing reasons.

Reversal of the final rejection of the above claim is respectfully requested.

B. Claim 44 was rejected under 35 U.S.C. § 103(a) as unpatentable over Mateos in view of Coates, and further in view of Chen (U.S. Patent No. 6,021,437).

1. Claim 44.

In view of the allowability of base claim 37 over Mateos and Coates, the obviousness rejection of dependent claim 44 over Mateos, Coates, and Chen has been overcome.

Moreover, claim 44 further recites that the data corresponding to the first embedded object comprises a current time and dynamic data gathered at the managed server at the current time. The Examiner conceded that Coates and Mateos fail to disclose the foregoing subject matter of claim 44, and instead, cited Chen as purportedly disclosing the claimed subject matter. 12/02/2009 Office Action at 12. The Examiner specifically cited to the following passages of Chen: Abstract, column 2, lines 31-67; column 4, lines 10-19. These passages of Chen refer to real-time monitoring of a data processing system. Chen, 2:34-35. However, there is no teaching or hint in Chen that any data that corresponds to an embedded object of a web page comprises both a **current time** and dynamic data gathered at the managed server **at the current time**.

Thus, even if Mateos, Coates, and Chen could be hypothetically combined, the hypothetical combination of the references would not have led to the subject matter of claim 44.

Reversal of the final rejection of the above claims is respectfully requested.

C. Claim 45 was rejected under 35 U.S.C. § 103(a) as unpatentable over Mateos in view of Coates, and further in view of Lynch (U.S. Patent No. 6,823,319).

1. Claim 45.

In view of the allowability of base claim 37 over Mateos and Coates, the obviousness rejection of dependent claim 45 over Mateos, Coates, and Lynch has been overcome.

Reversal of the final rejection of the above claim is respectfully requested.

CONCLUSION

In view of the foregoing, reversal of all final rejections and allowance of all pending claims is respectfully requested.

Respectfully submitted,

Date: April 30, 2010

/Dan C. Hu/

Dan C. Hu
Registration No. 40,025
TROP, PRUNER & HU, P.C.
1616 South Voss Road, Suite 750
Houston, TX 77057-2631
Telephone: (713) 468-8880
Facsimile: (713) 468-8883

VIII. APPENDIX OF APPEALED CLAIMS

The claims on appeal are:

29. A method of serving data from a management module of a managed server, comprising:

serving a web page to a requesting computer from the managed server, the web page comprising a source call to an object file and code including scripting functions defined by the object file, wherein the requesting computer is remote from the managed server, and wherein at least one of the scripting functions is for merging data associated with the object file with the web page;

receiving a request from the requesting computer at the managed server for the object file, wherein the request is received after the web page has been served to the requesting computer and after the requesting computer has evaluated the at least one scripting function;

populating the object file in real-time with data from the management module of the managed server after both serving the web page and receiving the request for the object file; and

serving the object file to the requesting computer after populating the object file.

30. The method of claim 29, wherein populating the object file comprises populating the object file with a scripting function.

31. The method of claim 30, wherein the scripting function populated in the object file is a JavaScript function.

32. The method of claim 29, wherein populating the object file comprises populating the object file with an array of data.

33. The method of claim 29, wherein populating the object file comprises acquiring real-time data indicative of a current status of a server.

34. The method of claim 29, wherein populating the object file comprises providing a language localization file.

35. The method of claim 29, wherein serving the web page comprises serving the web page configured for a handheld or palmtop computing platform.

36. The method of claim 29, wherein serving the web page comprises serving the web page across a firewall.

37. A method of displaying a web page, comprising:

requesting at least a frame of a web page from a managed server, wherein the frame comprises a first embedded object and a call to a scripting language function defined by the first embedded object, wherein the scripting language function is for merging data corresponding to the first embedded object with the web page;

receiving the frame from the managed server;

based on evaluating the scripting language function, requesting, by a requesting computer, the data corresponding to the first embedded object from the managed server after receiving the frame from the managed server;

receiving, by the requesting computer, the data corresponding to the first embedded object;

calling, by the requesting computer, the scripting language function defined by the first embedded object; and

merging, by the requesting computer, the data corresponding to the first embedded object into the frame.

38. The method of claim 37, further comprising displaying the frame.

39. The method of claim 37, further comprising evaluating the frame to identify a source tag of the first embedded object.

40. The method of claim 37, wherein the data corresponding to the first embedded object comprises dynamic data from a management module of the managed server.

41. The method of claim 40, wherein the dynamic data is generated in real-time in response to the request for the data corresponding to the first embedded object.

42. The method of claim 37, wherein the data corresponding to the first embedded object comprises a scripting language function.

43. The method of claim 42, wherein the frame comprises a second embedded object linked to dynamic data in the managed server, and wherein the scripting language function is configured to merge the dynamic data with the frame.

44. The method of claim 37, wherein the data corresponding to the first embedded object comprises a current time and dynamic data gathered at the managed server at the current time.

45. The method of claim 37, wherein merging the data comprises populating a drop-down menu with menu items.

46. A server, comprising:

a management module configured to generate dynamic data; and

a file system configured to store a web page that has both a first embedded object configured to access the dynamic data and a second embedded object configured to merge the dynamic data with the web page, wherein the first embedded object is executable on a client remote from the server to request the dynamic data from the server, and wherein the web page includes a scripting language function defined by the second embedded object, the scripting language function for merging the dynamic data with the web page;

wherein the server is configured to further:

send, to the client, the web page that has the first embedded object, the second embedded object, and the scripting language function defined by the second embedded object;

after sending the web page, receive a request from the client that is based on evaluating the scripting language function of the web page at the client;

in response to the request, retrieve the dynamic data and send the retrieved dynamic data to the client for merging with the web page.

47. The server of claim 46, wherein the second embedded object is executable on the client remote from the server to merge the dynamic data with the web page.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.